Interactive Restaurant Search System

Jannatul Ferdows[1]

1204091

Hoor A Zannat Rafcy[2]

1304062

January 1, 2017

Rahma Bintey Mufiz Mukta Farzana Yasmin

Dept. of Computer Science and Engineering
CUET

[1]E-mail address: kfoozminus@gmail.com
[2]E-mail address: hoora14zannat@gmail.com

0.1 Acknowledgement

We owe our deepest gratitude to the Department of Computer Science and Engineering, Cuet Campus for providing us with an opportunity to work on a Project as a part of our syllabus. It is an honor for us to express our gratitude to our project supervisor Rahma Bintey Mufiz Mukta and Farzana Yasmin for their constant support and guidance throughout the project . We are thankful to Rahma Bintey Mufiz Mukta and Farzana Yasmin for providing the constant feedbacks throughout the development of this report. Their suggestion throughout the development of this project helped us to cope up with different obstacles.

We are grateful to all our teachers for their suggestions and inspirational lectures that paved the towards the completion of this project .

Last but not the least, we would like to thank our classmate for their valuable comments and their suggestion during this project. Any kind of suggestion or criticism will be highly appreciated and acknowledged.

0.2 Abstract

Our project Interactive Restaurant Search System is an GPS based android application which helps the user to locate the restaurants nearby the current location and order the items available in the respective hotels and restaurant. The application tracks the current location of the user through the Google maps Api and it provides the nearest restaurants available from that location on the basis of ascending order of the distance .The apps shows the restaurants within 3 Km distance from the current location in the default screen. The user can view the menu and order the available items through this application. User can also view the restaurant location in the map and find the distance and route to the particular restaurant selected. This android based app for an Interactive Restaurant Search system allows users to find their desired food and beverage within seconds even in the middle of a busy street. This app helps the users to pick the proper restaurant instantly with the help of its search options. It will help them specially on sudden plan when someone dont know what to or where to eat so that they dont feel like wasting their money on something they never intended to have. This projects overall gives the easier tool for searching the restaurants available nearby and saves time with its simple and effective. This project work is primarily designed to give an insight to computer based interactive restaurant search system. It is as a result of problem associated with the existing system which involves the use of manual method in keeping information in the system. So among the numerous problems associated with the existing system are; staff are spending far too much time chasing mistakes instead of tending to customers, sales going unrecorded, inventory doesnt match your tallies and other. Computerized management information system database information system used by restaurant personnel to collect data, process it and also store it for future use. Researcher used visual basic in designing the system and Microsoft office as the database system. Interactive Restaurant Search System is an android based application primarily for use in the food delivery industry. This system will allow hotels and restaurants to increase scope of business by reducing the labor cost involved. The system also allows to quickly and easily manages an online menu which customers can browse and use to place orders with just few clicks. Restaurant employees then use these orders through an easy to navigate graphical interface for efficient processing.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The ubiquity of wireless networking and the trend toward component miniaturization have led to the evolution of cell phones from mere telephony devices to powerful mobile computing platforms that provide the basis for a host of other applications. Today's mobile phones are typically equipped with devices such as GPS sensors [1], Wi-Fi [2] and 3G wireless radios [3] capable streaming high bandwidth Internet content, touch-screen-based user interfaces [4], still and video cameras, Bluetooth transceivers [5], and Accelerometers [6]. Similar to a computer, a mobile operating system provides the primary execution environment for applications on the phone. Analogous to programs on a PC, apps can be downloaded and installed on mobile phones. Because of the growing general purpose computing capabilities of mobile devices, combined with their increasing popularity and adoption rate, it is expected that hand-held mobile phones will become the next PC. These technology trends have enabled innovative, exciting and compelling mobile applications to become widely available, from gaming to multimedia to social networking. Hand-in-hand with the growth of the raw computing power of mobile phones, various middleware/OS platforms have evolved that allow developers to take advantage of the computing resources to create feature-rich applications that provide compelling user interfaces and functionality. A wide selection of proprietary and open source mobile OS platforms exist, the most prominent ones being: Apple's iOS, Google's Android, Symbian from Symbian Foundation, RIM Blackberry OS, and Microsoft's Windows Mobile. Smartphone has got an integral part of our lives for several reasons: they bring our personal data, contacts, message, personal photos, bank account number and a variety of mobile applications that users are used to facilitate their daily life. It is known that a lots of people dont let off smart phones from their hands until they fall in a sleep. Thats the reason why mobile application for restaurant searching are more interesting and has become each day more important for restaurant marketing. The fact is the unfortunately, still a lots of restaurants dont have their own mobile application, or even the website. This is mainly because such investments are too expensive for them on their owners, consider them to be unnecessary. And unfortunately, that means that they lose a good deal of work and profits. We must realize that our potential customers become increasingly using their mobile devices seeking for location, information about restaurants when they want to have lunch or

drink at an unknown place. People want to save time, and in the simplest and quickest way possible to get to the information they are interested in. This application provides them exactly that luxury. It is cost-effective and incredibly important to increase sales and gaining new customers for new restaurants. This application is great enough for tourists and business people who travel frequently. Through it, people have to search instantly for a restaurant by category, location, special-food item or rating. This application is designed exclusively for the presentation and evaluation of restaurants. People love to comments about the restaurant which they want to visit and check that comments of guests who were there. By using this application people can make a comment and give ratings about restaurants offer, service, food-quality, taste, hygiene. All these comments and ratings are more efficient for people, who will make a choice and choose one restaurant after read the reviews of others similar hospitality facilities in a specific area. We think it is the best mobile application for food lovers whose desire is to get all the information about nearby restaurant instantly without wasting their valuable time.

1.2 Birth of an android

Android is a mobile phone operating system. It was originally developed by Android Inc, which was acquired by Google in July 2005. Today, development is overseen by the Android Open Source Project (AOSP), led by Google. The AOSP is tasked with the maintenance and further development of Android.

As of the 3rd Quarter of 2010 Android has a market share of 25% making it the second most popular phone operating system of the market (second only to Nokia's Symbian). This is a major rise from the 3.5% share Android had in 2009.

Android, Inc. was founded in Palo Alto, California, United States in October 2003 by Andy Rubin (co-founder of Danger), Rich Miner (co-founder of Wildfire Communications, Inc.), Nick Sears (once VP at T-Mobile),and Chris White (headed design and interface development at Web-TV)to develop, in Rubin's words "...smarter mobile devices that are more aware of its owner's location and preferences. Despite the obvious past accomplishments of the founders and early employees, Android Inc. operated secretly, revealing only that it was working on software for mobile phones. That same year, Rubin ran out of money. Steve Perlman, a close friend of Rubin, brought him 10,000 dollar in cash in an envelope and refused a stake in the company. Google acquired Android Inc. on August 17, 2005, making Android Inc. a wholly owned subsidiary of Google. Key employees of Android Inc., including Andy Rubin, Rich Miner and Chris White, stayed at the company after the acquisition.

1.3 Problem area

Restaurant is a kind of business that serves people all over world with ready-made food. Currently this industry is going on with lot of flare. People feel more comfortable with lot of variations in the selection and consumption of their food in their busy life. In

Figure 1.1: An android device

traditional booking system, a customer has to make a phone call in order to get his meal reserved. If luckily the phone gets connected, then the customer does some formal conversation like hello, hi, etc. Then he demands for todays menu and do some discussion over menu items then he orders and he has to give some of this identification specifications. This process takes 5-8 minutes to complete. On the receiver side there is hardly one phone line and one operator. So he can cover around 15-20 orders maximum in an hour. For each booking he has to register manually on paper and puts the order in a queue with specific priority according to time and quantity, and then a cook is assigned for the specific order to complete it. There are lots of areas to be solved for current restaurants using modern IT World. Many areas come like human resource management, accounts management, etc. But our problem lies within domain of end customer and restaurant Interactive Restaurant Search System.

1.4 Problem analysis

As discussed earlier our main problem area focuses on the Interactive restaurant Search System, there are lot of problems in that area which are associated with both the customer and the restaurant staff. We would like to analyze some of the problems here:

- Initial problem is that the customer has to get connected over the phone, it would be harder if the restaurant is very popular and busy.
- As customer wont have the menu list with him, it would be harder for him to remember the entire list (with price as well...!) and come to a decision, i.e. customer is provided with less time to make decision.

- The chances of committing mistakes at the restaurant side in providing a menu list for a specific time would be more.
- There might be some communication problems or sometimes language might be a barrier.
- As entire booking has to be done manually at the restaurant end, the chances of occurrence of mistakes is high as well.
- Most of restaurants have single phone line and a single operator to handle incoming calls, so they can accept limited orders.
- If the restaurant is of busy type, than the operator is left with no time to decide over the priority of the order fulfillment. item Even assigning orders (or some menu from the order) to a specific cook can be cumbersome if it is done parallel with the bookings of the order.
- All the calls will not by intended for booking, as some calls might be for canceling the order or to fetch the status as well, this eats up the productive time at the restaurant side.

1.5 Why we will use this app

We have seen that we will face different types of problem to order an item by phone call. All these problems will be solved by using our application. Some advantages of using our application are below

- This android based app for an Interactive Restaurant Search System allows users to find their desired food or beverage within seconds even in the middle of a busy street.
- This app helps the users to pick the proper restaurant instantly with the help of its search options. It will help them specially on sudden plan when someone dont know what to or where to eat so that they dont feel like wasting their money on something they never intended to have.
- All restaurant authority will require signing up so that they can get notification regarding order.
- Users dont have to sign up for this separately, one can easily start using it in any moment, anyplace.
- A user can choose their restaurant by one of the three search options
    1. Search by Restaurant Name: If the user already picked his restaurant, this app can show its location, all their menus with price and photo and rating.
    2. Search by Item Name or Category: It retrieves a list of items that contains the search word in its title or the category it is included. This result list can be sorted with respect to their relevance or rating or the location of their corresponding restaurant.

3. Search by Location: After turning on location in their android device, nearest places which would be perfect is suggested, based on collaborative priorities of rating and distance.

- As soon as user starts typing the keyword for the search, the app will fetch and show the names of the restaurants whose name starts with the letters already typed, from which user can select their desired one.

- After finding the suitable restaurant, user can order meals or book tables by clicking on 'order' or 'Book a Table' and the according restaurant will get notification containing the location and order of the consumer.

- User can also book a table or order meals by contacting them over the phone. To make that easier, contact number of that restaurant will appear on the screen as soon as user clicks on 'Contact.

- User can also view the entire database of the restaurant sorted lexicographically.

- If a user wants to rate a restaurant or a menu or requesting a new directory, that is, to say, contributing to the entire database, that would require signing up. Here, requesting a new directory or to change information of a restaurant will go through confirmation.

- It allows a business owner to leverage the performance of staff and reduce organizational costs. There is always a possibility to integrate it with an existing IT solution that is already used in order to achieve a very flexible combination with a lot of advantages. For example, when it is necessary to change the cost or the names of several dishes, add or remove menu items there is no need to input a stock list twice.

- The application is really great as it can always be at hand informing the users about the latest news. The app is very user-oriented and offers deals according to personal preferences of the user and their behavior. It also provides information about new products, promotions and nutritional facts. Users can search for the nearest restaurant and review details about it.

- The main advantage of my system is that it greatly simplifies the ordering process for both the customer and the restaurant and also greatly lightens the load on the restaurants end, as the entire process of taking orders is automated.

Figure 1.2: Figure 1

Table 1.1: Table 1

Chapter 2

Literature review

2.1 Language

- Java
- PHP
- MYSQL

2.2 Links

We took help from these website on these dates-

- 27 november - w3school.
- 12 December - https://www.youtube.com/watch?v=uzeP59KnbPg
- 13 December - https://www.youtube.com/watch?v=_7r_vdwmW0o
- 15 December - ttps://www.simplifiedcoding.net/android-json-tutorial-to-get-data-from-mysql-database.
- 26 December - http://kosalgeek.com/connect-android-php-mysql-generic-asynctask
- 27 December - https://goo.gl/sk5otF

Chapter 3

E-R Diagram

3.1 Introduction

An entity-relationship diagram is usually the result of systematic analysis to define and describe what is important to processes in an area of a business. It does not define the business processes; it only presents a business data schema in graphical form. It is usually drawn in a graphical form as entities that are connected by lines which express the associations and dependencies between entities. An ER diagram can also be expressed in a verbal form, for example: one building maybe divided into zero or more apartments, but one apartment can only be located in one building. Entities maybe characterized not only by relationships, but also by additional properties, which include identifiers called primary key. Diagrams created to represent attributes as well as entities and relationships maybe called entity-attribute-relationship diagrams, rather than entity-relationship models. An ER model is typically implemented as a database. In a simple relational database implementation, each row of a table represents one instance of an entity type, and each field in a table represents an attribute type. In a relational database a relationship between entities is implemented by storing the primary key of one entity as a pointer or foreign key in the table of another entity. There is a tradition for ER/data models to be built at two or three levels of abstraction. Note that the conceptual-logical-physical hierarchy below is used in other kinds of specification, and is different from the three schema approach to software engineering.

3.1.1 Conceptual Model

This is the highest level ER model in that it contains the least granular detail but establishes the overall scope of what is to be included within the model set. The conceptual ER model normally defines master reference data entities that are commonly used by the organization. Developing an enterprise-wide conceptual ER model is useful to support documenting the data architecture for an organization. A Conceptual ER model maybe used as the foundation for one or more logical data models. The purpose of the conceptual ER model is then to establish structural meta-data commonality for the master data entities between the set of logical ER models. The conceptual data model maybe used to form commonality relationships between ER models as a basis for data model integration.

3.1.2 Logical Model

A logical ER model does not require a conceptual ER model, especially if the scope of the logical ER model includes only the development of a distinct information system. The logical ER model contains more details than the conceptual ER model. In addition to master data entities are now defined. The details of each data entity are developed and the relationships between these data entities are established. The logical ER model is however developed independently of the specific database management system into which it can be implemented.

3.1.3 Physical Model

One or more physical ER models maybe developed from each logical ER models. The physical ER model is normally developed to be instantiated as a database. Therefore, each physical ER model must contain enough physical ER model is technology dependent since each database management system is somewhat different. The physical model is normally instantiated in the structural meta-data of a database management system as relational database objects such as database tables, database indexes such as a unique key indexes, and database constraints such as a foreign key constraints or a commonality constraint. The ER model is also normally used o design modifications to the relational database objects and to maintain the structural meta-data of the database.

The first stage of information system design uses these models during the requirements analysis to describe information needs or the type of information that is to be stored in a 0database. The data modeling technique can be used to describe any ontology for a certain area of interest. In the case of the design of an information system that is based on a database, the conceptual data model is, at a later stage(usually called logical design), mapped to a logical data model, such as the relational model, this in turn is mapped to a physical model during physical design. Note that sometimes, both of these phases are referred to as physical design .

3.2 E-R Diagram Notation

An ER diagram consists of the following major components:

1. Rectangles divided into two parts represent entity sets. The first part contains the name of the entity set. The second part contains the names of all the attributes of the entity set.

2. Diamonds represent relationship sets.

3. Undivided rectangles represent the attributes of a relationship set. Attributes that are part of the primary key are underlined.

4. Lines link entity sets to relationship sets.

5. Dashed lines link attributes of a relationship set to the relationship set.

6. Double lines indicate total participation of an entity in a relationship set.

7. Double diamonds represent identifying relationship sets linked to weak entity sets

3.3 Entity Sets

An entity is a thing or object in the real world that is distinguishable from all other objects. For example, each person in a university is an entity. An entity has a set of properties, and the values for some set of properties may uniquely identify an entity. For instance, a person may have a `person_id` property whose value uniquely identifies that person. Thus, the value 8099023 for `person_id` would uniquely identify one particular person in the university. Similarly, courses can be thought of as entity, and`course_id` uniquely identifies a course entity in the university. An entity may be concrete, such as a person or a book, or it may be abstract, such as a course, a course offering, or a flight reservation. An entity set is a set of entities of the same type that share the same properties, or attributes. The set of all people who are instructors at a given university, for example, can be defined as an entity set instructor. Similarly, the entity set student might represent the set of all students in the university. In the process of modeling, we often use the term entity set in the abstract, without referring a particular set of individual entities. We use the term extension of the entity set to refer to the actual collection of entities belonging to the entity set. Thus, the set of actual instructors in the university forms the extension of the entity set instructor. Entity sets do not need to disjoint. For example, it is possible to define the entity set of all people in a university. A person entity may be an instructor entity, a student entity, both, or neither. An entity is represented by a set of attributes. Attributes are descriptive properties possessed by each member of an entity set. The designation of an attribute for an entity set expresses that the database stores similar information concerning each entity in the entity set; however, each entity may have its own value for each attribute.

3.3.1 Strong Entity

Most of the basis entity types to identify in an organization are classified as strong entity types. A strong entity type is one that exists independently of other entity types. Instances of a strong entity type always have a unique characteristic that is, an attribute or combination of attributes that uniquely distinguish each occurrence of that entity.

3.3.2 Weak entity

A weak entity type is an entity type whose existence depends on some other entity type. A weak entity type has no business meaning in the ER diagram without the entity on which it depends. The entity type on which the weak entity type depends is called the identifying owner. A weak entity type does not have its own identifier. Generally on an ER diagram a weak entity type has an attribute that serves as a partial identifier. During a later design stage a full identifier will be formed for the weak entity by combining the partial identifier with the identifier of its owner.

3.3.3 Associative Entity

The presence of one or more attributes on a relationship suggests to the designer that the relationship should perhaps instead be represented as an entity type. An associative entity is an entity type that associates the instances of one or more entity types and contains attributes that are peculiar to the relationship between those entity instances. The purpose of this special symbol is to preserve the information that the entity was initially specified as a relationship on the ER diagram. Associative entities are sometimes referred to as gerunds, since the relationship name is usually converted to an entity name that is a noun.

3.4 Relationship Sets

A relationship is an association among several entities. A relationship instance in an ER schema represents an association between the named entities in the real world enterprise that is being modeled. The function that an entity plays in a relationship is called that entitys role. Since entity sets participating in a relationship set are generally distinct, roles are implicit and are not usually specified. However, they are useful when the meaning of a relationship needs clarification. Such is the case when the entity sets of a relationship set are not distinct; that is, the same entity set participates in a relationship set more than once, in different roles. In this type of relationship set, sometimes called a recursive relationship set, explicit role names are necessary to specify how an entity participates in a relationship instances. A relationship may also have attributes called descriptive attributes. A relationship instance in a given relationship set must be uniquely identifiable from its participating entities, without using the descriptive attributes. The number of entity sets that participates in a relationship set is the degree of the relationship set. A binary relationship set is of degree 2; a ternary relationship set is of degree 3

3.5 Attributes

For each attribute, there is a set of permitted values, called the domain, or value set, of that attribute. The domain of attribute `course_id` might be the set of all text strings of a certain length. Formally, an attribute of an entity set is a function that maps from the entity set into a domain. Since an entity set may have several attributes, each entity can be described by a set (attribute, data value) pairs, one pair for each attribute of the entity set. The attribute values describing an entity constitute a significant portion of the data stored in the database. An attribute, as used in the ER model, can be characterized by the following attribute types.

3.5.1 Simple Attribute

A simple (or atomic) attribute is an attribute that cannot be broken down into smaller components. For example, all of the attributes associated with automobile are simple: Vehicle ID, Color, Weight, and Horsepower.

3.5.2 Composite Attribute

These attributes can be divided into sub-parts. For example, an attribute name could be structured as a composite attribute consisting of `first_name`,`middle_name`, and `last_name`. Using composite attribute in a design schema is a good choice if a user will wish to refer to an entire attribute on some occasions, and to only a component of the attribute on other occasions. Composite attributes help us to group together related attributes, making the modeling clearer.

3.5.3 Single-Valued Attribute

These attribute have a single value for a particular entity. For instance, the `student_ID` attribute for a specific student entity refers to only one `student_ID`. Such attribute are said to be single valued attribute.

3.5.4 Multi-Valued Attribute

A multi-valued attribute is an attribute that may take on more than one value for a given entity instance. We indicate a multi-valued attributed with an ellipse with double lines.

3.5.5 Derived Attribute

Some attribute values that are of interest to users can be calculated or derived from other related attribute values that are stored in the database. For example, suppose that for an organization, the EMPLOYEE entity type has a `Date_Employed` attribute. If users need to know how many years a person has been employed, that value can be calculated using `Date_Employed` and todays date. A derived attribute is an attribute whose values can be calculated from related attribute values. We indicate a derived attribute in an ER diagram by using an ellipse with a dashed line. In some situations the value of an attribute can be derived from attribute in related entities.

3.5.6 Identifier Attribute

An identifier is an attribute that uniquely identifies individual instances of an entity type. The identifier for the STUDENT entity type introduced earlier is `STUDENT_ID`, while the identifier for the AUTOMOBILE is `Vehicle_ID`. Notice that an attribute such `Student_Name` is not a candidate identifier, since many students may potentially have the same name and students, like all people, can change their names. To be a candidate identifier, each entity instance must have a single value for the attribute names on the ER diagram. For some entity types, there is no single attribute that can serve as the identifier. However, two attributes used in combination may serve as the identifier. A composite attribute is an identifier that consists of a composite attribute. Some entities may have more than one candidate attribute.

3.6 Degree of a relationship

The degree of a relationship is the number of entity types that participate in that relationship. The three most common relationship degrees in ER models are unary(degree 1), binary(degree 2) and ternary(degree 3). Higher-degree relationships are possible, but they are rarely encountered in practice, so we restrict our discussion to these three cases.

3.6.1 Unary Relationship

A unary relationship is a relationship between the instances of a single entity type. It is also called recursive relationship.

3.6.2 Binary Relationship

A binary relationship is a relationship between the instances of two entity types and is the most common type of relationship encountered in data modeling. The first indicates that an employee is assigned one parking place, and each parking place is assigned to one employee. The second indicates that a product line may contain several products and each product belongs to only one product line. The third shows that a student may register for more than one course, and that each course may have many student registrants.

3.6.3 Ternary Relationship

A ternary relationship is a simultaneous relationship among the instances of three entity types. It is not the same as three binary relationships.

3.7 Cardinality Constraints

The relationship set advisor, between the instructor and student entity sets may be one-to-one, one-to-many, many-to-one, many-to-many. To distinguish among these types, we draw either a directed line or an undirected line between the relationship set and the entity set as follows:

3.7.1 Minimum Cardinality

The minimum number of instances of one entity that may be associated with each instance of another entity is called minimum cardinality.

3.7.2 Maximum Cardinality

The maximum number of instances of one entity that may be associated with each instance of another entity is called maximum cardinality.

In our ER diagram Restaurant,User and order are strong entity.Item is weak entity because it depends on Restaurant.In Restaurant,Trade-license-no is a primary key and Delivery-fee,Name,Location,Rating,Opening-hours,No-of-Users,is its attributes.Here,Location and COntact-no are multi-valued attributes.Location and Opening-Hours are composite attribute.For user,Contact-no,Name,Login-id,Password,User-type are attributes and Login-Id is a primary key.Order has 3 attributes-Order-No, Is-delivered and Total-Price. Item has 5 attributes.They are-No-Of-Users-Rated,Price,Rating,Name,Image.Name is a primary key of weak entity Item.

3.8 Conclusion

ER assumes information content that can readily be represented in a relational database. They describe only a relational structure for this information.They are inadequate for systems in which the information cannot readily be represented in relational map, such as with semi-structured data.For many systems, possible changes to information contained are nontrivial and important enough to warrant explicit specification.Some authors have extended ER modeling with constructs to represent change, an approach supported by the original author an example is Anchor Modeling. An alternative is to model change separately, using a process modeling technique. Additional techniques can be used for other aspects of systems. Even where it is suitable in principle, ER modeling is rarely used as a separate activity. One reason for this is today's abundance of tools to support diagramming and other design support directly on relational database management systems. These tools can readily extract database diagrams that are very close to ER diagrams from existing databases, and they provide alternative views on the information contained in such diagrams.The enhanced entity relationship model (EER modeling) introduces several concepts not in ER modeling, but are closely related to object-oriented design, like is a relationships.For modeling temporal databases, numerous ER extensions have been considered.[21] Similarly, the ER model was found unsuitable for multi-dimensional database.

Chapter 4

Relational Mapping

4.1 Introduction

Transforming (or mapping) ER diagram to relations is a relatively straight forward process with a well defined set of rules. In fact, many CASE tools can automatically perform many of the conversation steps. However, it is important that we understand the steps in this process for three reasons:

1. CASE tools often cannot model more complex data relationships such as ternary relationships and super-type/sub-type relationships. For these situations we may have to perform the steps manually.
2. There are sometimes legitimate alternatives where we will need to choose a particular solution.
3. We must be prepared to perform a quality check on the results obtained with a CASE tool.

We are familiar to three types of entities. They are:

4.1.1 Regular entities

They are entities that have an independent existence and generally represent real-world objects such as persons and products. Regular entity types are represented by rectangles with a single line.

4.1.2 Weak entities

They are entities that cannot exist except with an identifying relationship with an owner entity type. Weak entities are identified by a rectangle with a double line.

4.1.3 Associative entities

They are also called gerunds. They are formed from many-to-many relationships between other entity types. Associative entities are represented by rectangle with a single line that encloses the diamond relationship symbol.

There are 7 steps to transform an ER diagram to relational mapping. They are below:

4.2 Steps 1

4.2.1 Mapping of regular entity types

1. For each regular entity type E in the ER schema, create a relation R that includes all the simple attributes of E.
2. Choose one of the key attributes of E as the primary key for R.
3. If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

4.3 Steps 2

4.3.1 Mapping of weak entity types:

1. For each weak entity W in the ER schema with owner entity type E, create a relation R and include all simple attributes (or simple components of composite attribute) of W as attributes of R.
2. Also, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s).
3. The primary key of R is the combination of the primary key(s) of the owner and the partial key of the weak entity type W, if any.

4.4 Steps 3

4.4.1 Mapping of Binary 1:1 relation types:

1. For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R.
2. There are three possible approaches:
    - Foreign Key approaches: Choose one of the relation say S and include a foreign key in S in the primary key of T. It is better to choose an entity type with total participation in R in the role of S.
    - Merged relation option: An alternate mapping of a 1:1 relationship type is possible by merging the two entity types and the relationship into a single relation. This may be appropriate when both participations are total.
    - Cross-reference or relationship relation option: The third alternative is to set up a third relation R for the purpose of cross- referencing the primary keys of the two relations S and T representing the entity types.

4.5 Steps 4

4.5.1 Mapping of binary 1: N relationship types:

1. For each regular binary 1: N relationship type R, identify the relation S that represent the participating entity type at the N side of the relationship type.

2. Include as foreign key in S the primary key of the relation T that represents other entity type participating in R.

3. Include any simple attributes of the 1: N relation type as attributes of S.

4.6 Steps 5

4.6.1 Mapping of binary M: N relationship types

1. For each regular binary M: N relationship type R, create a new relation S to represent R.

2. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S.

3. Also include any simple attributes of the M: n relationship type(or simple components of composite attributes) as attributes of S.

4.7 Steps 6

4.7.1 Mapping of multi-valued attributes

1. For each multi-valued attribute A, create a new relation R.

2. This relation R will include an attribute corresponding to A, plus the primary key attribute K as a foreign key in R of the relation that represents the entity type of relationship type that has A as an attribute.

3. The primary key of R is the combination of A and K. If the multi-valued attribute is composite, we include its simple components.

4.8 Steps 7

4.8.1 Mapping of N-ary relationship types

1. For each n-ary relationship type R, where n>2, create a new relationship S to represent R.

2. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.

3. Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.

4.9 Conclusion

Relational mapping (mapping tool) in computer science is a programming technique for converting data between incompatible type systems in programming languages. This creates, in effect, a "virtual object database" that can be used from within the programming language. There are both free and commercial packages available that perform Relational mapping, although some programmers opt to construct their own ORM tools.

Data-management tasks act on object-oriented (OO) objects that are almost always non-scalar values. For example, consider an address book entry that represents a single person along with zero or more phone numbers and zero or more addresses. This could be modeled in an object-oriented implementation by a "Person object" with attributes/fields to hold each data item that the entry comprises: the person's name, a list of phone numbers, and a list of addresses. The list of phone numbers would itself contain "Phone-Number objects" and so on. The address-book entry is treated as a single object by the programming language (it can be referenced by a single variable containing a pointer to the object, for instance). Various methods can be associated with the object, such as a method to return the preferred phone number, the home address, and so on.

However, many popular database products such as SQL database management systems (DBMS) can only store and manipulate scalar values such as integers and strings organized within tables. The programmer must either convert the object values into groups of simpler values for storage in the database (and convert them back upon retrieval), or only use simple scalar values within the program. Object-relational mapping implements the first approach.

The heart of the problem involves translating the logical representation of the objects into an atomized form that is capable of being stored in the database, while preserving the properties of the objects and their relationships so that they can be reloaded as objects when needed. If this storage and retrieval functionality is implemented, the objects are said to be persistent.

Chapter 5

Normalization

5.1 Introduction

Normalization is a formal process for deciding which attributes should be grouped together in a relation. Before proceeding with physical design, we need a method to validate the logical design to this point. Normalization is primarily a tool to validate and improve a logical design, so that it satisfies certain constraints that avoid unnecessary duplication of data. We have presented an intuitive discussion of well structured relations; however, we need formal definitions of such relations, together with a process for designing them. Normalization is the process of decomposing relations with anomalies to produce smaller, well-structured relations. Normalization involves arranging attributes in tables based on dependencies between attributes, ensuring that the dependencies are properly enforced by database integrity constraints. Normalization is accomplished through applying some formal rules either by a process of synthesis or decomposition. Synthesis creates a normalized database design based on a known set of dependencies. Decomposition takes an existing (insufficiently normalized) database design and improves it based on the known set of dependencies. Edgar F. Codd, the inventor of the relational model (RM), introduced the concept of normalization and what we now know as the First normal form(1NF) in 1970.Codd went on to define the Second normal form (2NF) and Third normal form (3NF) in 1971, and Codd defined the Boyce-Codd Normal Form (BCNF) in 1974. Informally, a relational database table is often described as "normalized" if it meets Third Normal Form.

5.2 Step of normalization

Normalization can be accomplished and understood in stages¡ each of which corresponds to a normal form. A normal form is a state of a relation that results from applying simple rules regarding functional dependencies to that relation. We describe these rules briefly in this chapter and illustrate them in details in the following sections.

1. First Normal Form: Any multi-valued attributes (also called repeating groups) have been removed, so there is a single value at the intersection of each row and column of the table.

2. Second Normal Form: Any partial functional dependencies have been removed.

3. Third Normal Form: Any transitive dependencies have been removed.

4. Boyce/Codd Normal Form: Any remaining anomalies that result from functional dependencies have been removed.

5. Fourth Normal Form: Any multi-valued dependencies have been removed.

6. Fifth Normal Form: Any remaining anomalies have been removed. We describe and illustrate first through third normal forms in our project.

5.3 Functional Dependencies

Normalization is based on the analysis of functional dependencies. A functional dependency is a constraint between two attributes or two sets of attribute A if, for every valid instance of A, that value of A uniquely determines the value of B. The functional dependency of B on A is represented by an arrow. An attribute may be functionally dependent on two attributes, rather than on a single attribute. Common examples of functional dependencies are the following:

1. SSN- (Name, Address, Birthdate)- A persons name, address, and birthdate are functionally dependent on that persons Social Security number.

2. VIN-(Make, Model, Color)- The make, model, and color of a vehicle are functionally dependent on the vehicle identification number.

3. ISBN-(Title, First-Author-Name)- The title of a book and the name of the first author are functionally dependent on the books International Standard Book Number.

5.4 Keys

Keys can be simple (a single field) or composite (more than one field).Keys usually are used as indexes to speed up the response to user queries. Keys are special fields that serve two main purposes:

1. Primary Keys: Primary Keys are uniquely identifiers of the relation. Examples include employee numbers, social security numbers etc. This is how we can guarantee that all rows are unique.

2. Foreign Keys: Foreign Keys are identifiers that enable a dependent relation (on the many side of a relationship) to refer to its parent relation (on the one side of the relationship)

3. Candidate Keys: A candidate key is an attribute, or combination of attributes that uniquely identifies a row in a relation.

A candidate key must satisfy the following properties, which are a subset of the six properties of a primary key:

(i) **Unique Identification** For every row, the value of the key must uniquely identify that row. This property implies that each non-key attribute is functionally dependent on that key.

(ii) **Non-redundancy** No attribute in the key can be deleted without destroying the property of unique identification.

Determinant

The attribute on the left-hand side of the arrow is a functional dependency is called a determinant. SSN, VIN, ISBN are determinants.

5.5 Basic Normal Form

We have examined functional dependencies and keys, we are ready to describe and illustrate first through third normal forms. We also describe the normalization of summary data that appear in information bases.

5.5.1 First Normal Form

A relation is in first normal form (1NF) if it contains no multi-valued attributes. Recall that the first property of a relation is that the value at the intersection of each row and column must be atomic. Thus a table that contains multi-valued attributes or repeating groups is not a relation. A table with multi-valued attributes is converted to a relation in first normal form by extending the data in each column to fill cells that are empty because of the multi-valued attributes. In the relational model, we formalize this idea that attributes do not have any substructure. A domain is atomic if elements of the domain are considered to be indivisible units. We say that a relation schema R is in first normal form (1NF) if the domains of all attributes of R are atomic. A set of names is an example of a non-atomic value. For example, if the schema of a relation employee included an attribute -children whose domain elements are sets of names, the schema would not be in first normal form. Composite attributes, such as an attribute address with component attributes - street, city, state, and zip also have non-atomic domains. Integers are assumed to be atomic, so the set of integers is an atomic domain; however, the set of all sets of integers is a non-atomic domain. The distinction is that we do not normally consider integers to have sub-parts, but we consider sets of integers to have sub-parts namely, the integers making up the set. But the important issue is not what the domain itself is, but rather how we use domain elements in our database. The domain of all integers would be non-atomic if we considered each integer to be an ordered list of digits. The use of set-valued attributes can lead to designs with redundant storage of data, which in turn can result in inconsistencies. For instance, instead of having the relationship between instructors and sections being represented as a separate relation teaches, a database designer may be tempted to store a set of

course section identifiers with each instructor and a set of instructor identifiers with each section. (The primary keys of section and instructor are used as identifiers.) Whenever data pertaining to which instructor teaches which section is changed, the update has to be performed at two places: in the set of instructors for the section, and the set of sections for the instructor. Failure to perform both updates can leave the database in an inconsistent state. Keeping only one of these sets, that either the set of instructors of a section, or the set of sections of an instructor, would avoid repeated information; however keeping only one of these would complicate some queries, and it is unclear which of the two to retain. Some types of non-atomic values can be useful, although they should be used with care. For example, composite-valued attributes are often useful, and set valued attributes are also useful in many cases, which is why both are supported in the E-R model. In many domains where entities have a complex structure, forcing a first normal form representation represents an unnecessary burden on the application programmer, who has to write code to convert data into atomic form. There is also the run-time overhead of converting data back and forth from the atomic form. Support for non-atomic values can thus be very useful in such domains. In fact, modern database systems do support many types of non-atomic values.

5.5.2 Second Normal Form

A relation is in second normal form (2NF) if it is in first normal form and every non-key attribute is fully functionally dependent on the primary key. Thus no non-key attribute is functionally dependent on part of the primary key. A relation that is in first normal form will be in second normal form if any one of the following conditions applies:

1. The primary key consists of only one attribute (such as the attribute `EmpID` in EMPLOYEE1).

2. No non-key attributes exist in the relation (thus all of the attributes in the relation are components of the primary key).

3. Every non-key attribute is functionally dependent on the full set of primary key attribute.

Partial Functional Dependency

A partial dependency is a functional dependency in which one or more non-key attributes are functionally dependent on part of the primary key. Decomposition Using Functional Dependencies:

We noted that there is a formal methodology for evaluating whether a relational schema should be decomposed. This methodology is based upon the concepts of keys and functional dependencies. In discussing algorithms for relational database design, we shall need to talk about arbitrary relations and their schema, rather than talking only about examples. Recalling our introduction to the relational model, we summarize our notation here. In general, we use Greek letters for sets of attributes (for example). We use a lowercase Roman letter followed by an uppercase Roman letter in parentheses to refer to a relation schema (for example, r(R)). We use the notation r(R) to show that

the schema is for relation r, with R denoting the set of attributes, but at times simplify our notation to use just R when the relation name does not matter to us. Of course, a relation schema is a set of attributes, but not all sets of attributes are schemas. When we use a lowercase Greek letter, we are referring to a Relational Database Design of attributes that may or may not be a schema. A Roman letter is used when we wish to indicate that the set of attributes is definitely a schema.

- When a set of attributes is a super key, we denote it by K. A super key pertains to a specific relation schema, so we use the terminology K is a super key of r(R).
- We use a lowercase name for relations. In our examples, these names are intended to be realistic (for example, instructor), while in our definitions and algorithms, we use single letters, like r.
- A relation, of course, has a particular value at any given time; we refer to that as an instance and use the term instance of r. When it is clear that we are talking about an instance, we may use simply the relation name (for example, r).

5.5.3 Third Normal Form

BCNF requires that all nontrivial dependencies be of the form , where is a super-key. Third normal form (3NF) relaxes this constraint slightly by allowing certain nontrivial functional dependencies whose left side is not a super-key. Before we define 3NF, we recall that a candidate key is a minimal super-key that is, a super-key no proper subset of which is also a super-key. A relation schema R is in third normal form with respect to a set F of functional dependencies if, for all functional dependencies in F + of the form , where R and R, at least one of the following holds: Note that the third condition above does not say that a single candidate key must contain all the attributes in ; each attribute A in may be contained in a different candidate key. The first two alternatives are the same as the two alternatives in the definition of BCNF. The third alternative of the 3NF definition seems rather unintuitive, and it is not obvious why it is useful. It represents, in some sense, a minimal relaxation of the BCNF conditions that helps ensure that every schema has a dependency preserving decomposition into 3NF. Its purpose will become more clear later, when we study decomposition into 3NF. Observe that any schema that satisfies BCNF also satisfies 3NF, since each of its functional dependencies would satisfy one of the first two alternatives. BCNF is therefore a more restrictive normal form than is 3NF. The definition of 3NF allows certain functional dependencies that are not allowed in BCNF. A dependency that satisfies only the third alternative of the 3NF definition is not allowed in BCNF, but is allowed in 3NF.

In our Normalization,the relational model has multi-valued attributes.Location and Contact-no are multi-valued.So to transform in a first normal form we have to change the multi-valued attributes as single valued.Then the relation model will be in first normal form.After creating first normal form,we have to check that they have any partial dependencies or not.If there exists partial dependencies then we have to remove these.But in our Diagram, there is no partial functional dependencies.So we don't need any change in our first normal form and it is second normal form.Then to change in third normal form we have to check if there is any transitive dependencies.But in our diagram, there is no transitive dependencies.So we don't need to change anything.And second normal form will be the third normal form.

5.5.4 Conclusion

Occasionally database designers choose a schema that has redundant information; that is, it is not normalized. They use the redundancy to improve performance for specific applications. The penalty paid for not using a normalized schema is the extra work (in terms of coding time and execution time) to keep redundant data consistent. For instance, suppose all course prerequisites have to be displayed along with a course information, every time a course is accessed. Functional dependencies can help us detect poor E-R design. If the generated relation schemas are not in desired normal form, the problem can be fixed in the ER diagram. That is, normalization can be done formally as part of data modeling. Alternatively, normalization can be left to the designers intuition during E-R modeling, and can be done formally on the relation schemas generated from the E-R model. A careful reader will have noted that in order for us to illustrate a need for multi-valued dependencies and fourth normal form, we had to begin with schema that were not derived from our E-R design. Indeed, the process of creating an E-R design tends to generate 4NF designs. In our normalized schema, this requires a join of course with prereq. One alternative to computing the join on the fly is to store a relation containing all the attributes of course and prereq. This makes displaying the full course information faster. However, the information for a course is repeated for every course prerequisite, and all copies must be updated by the application, whenever a course prerequisite is added or dropped. The process of taking a normalized schema and making it non-normalized is called de-normalization, and designers use it to tune performance of systems to support time-critical operations. A better alternative, supported by many database systems today, is to use the normalized schema, and additionally store the join of course and prereq as a materialized view. (Recall that a materialized view is a view whose result is stored in the database and brought up to date when the relations used in the view are updated) Like de-normalization, using materialized views does have space and Relational Database Design time overhead; however, it has the advantage that keeping the view up to date is the job of the database system, not the application programmer.

Chapter 6

Implementation

6.1 Requirement Analysis

Requirement analysis is the first thing to be done in starting any kind of project. Requirement analysis is important and preliminary factor of software development process. It helps to identify the feasibility of project and many more. Hence in order to make our project real project and to realize the project in a practical way, we first analyze the requirement of our project. Specifically the requirements of our project are tabulated as:

- Identify the Current location of user,
- List out the nearest Restaurants on the basis of distance,
- User should be able to order the items from the hotel.
- User should be guided to the hotel through the map and route the path.

6.2 Methodology And Tools

Java is used as programming language to implement and test the application. In this projects of GPS based android system ,we have used the java programming language and the IDE used is the Android Studio. Before starting the android programming in Android Studio, We need the components like JDK, Android software development kit (SDK), Android Development Tools (ADT) plugin for

The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. The officially supported integrated development environment (IDE) is Eclipse using the Android Development Tools (ADT) Plugin, though IntelliJ IDEA IDE (all editions) fully supports Android development out of the box, and NetBeans IDE also supports Android development via a plugin. Also SDK provides several interface and drag and drop features in it. We have used several features available in it as per our need and the requirement.

6.3 Background Theories

One of the defining features of mobile phones is their portability, so it's not surprising that someof the most enticing Android features are the services that let us find, contextualize, and map physical locations

6.3.1 Location-Based-Service

Location-Based Services (LBS) the services that let us find the devices current location. They include technologies like GPS and Googles cell-based location technology. We can specify which location-sensing technology to use explicitly by name, or implicitly by defining a set of criteria in terms of accuracy, cost, and other requirements. Google map API has the in built library through which we can find the device current location .Further it has the ability to listen the location change of the device. Whenever the location of the device is changed from a particular location, it provides the notification of location changed through the features available in it. Location-based services (LBS) is an umbrella term used to describe the different technologies used to find the device's current location. The two main LBS elements are:

1. Location Manager: Provides hooks to the location-based services.
2. Location Providers: Each of which represents a different location-finding technology used to determine the device's current location.

Using the Location Manager, we can:

- Obtain your current location.
- Track movement.
- Set proximity alerts for detecting movement into and out of a specified area.

Location-based services use latitude and longitude to pinpoint geographic locations,but users are more likely to think in terms of an address.Location-based services provide a powerful toolkit for incorporating phone's native mobility into mobile applications.

6.3.2 MySQL Database

MySQL is primarily an RDBMS and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use.

6.3.3 JavaScript Object Notation

JSON (JavaScript Object Notation) is a text-based open standard designed for human readable independent data interchange. JSON is limited to text and numeric values. Binary values are not supported. JSON is a subset of the JavaScript Specification (ECME-Script) and it is therefore directly supported in JavaScript. Data structures in JSON are based on key / value pairs. The key is a string, the value can be a numerical value, a boolean value (true or false) or an object. The JSON format is often used for serializing and transmitting structured data over a network connection. It is used primarily to transmit data between a server and web application, serving as an alternative to XML.

6.3.4 JSON Datatypes And Syntex

JSON's basic types are:

- Number (double precision floating-point format in JavaScript, generally depends on implementation)
- String (double-quoted Unicode, with backslash escaping)
- Boolean (true or false)
- Array (an ordered sequence of values, comma-separated and enclosed in square brackets; the values do not need to be of the same type)
- Object (an un-ordered collection of key:value pairs with the ':' character separating the key and the value, comma-separated and enclosed in curly braces; the keys must be strings and should be distinct from each other)
- null empty

An JSON object is a set of key / value pairs which starts with "{" and ends with "}".

```
{
{
firstName:'Lars',
lastName:'Vogel',
address: {
street:'Examplestr.',
number: '31'
}
}

JSON arrays are one or more values surrounded by [] and separated by ",".

[
{
firstName:'Lars',
```

```
lastName:'Vogel',
address: {
street:'Examplestr.',
number: '31'
}
}
,
{
firstName:'Jack',
lastName:'Hack',
address: {
street:'Examplestr.',
number: '31'
}
}
]
```

6.3.5 Parsing through RESTful Services And JSON Response

Roy Fielding's explanation of the meaning of REST was "Representational State Transfer parallel is intended to evoke an image of how a well-designed Web application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing the next state of the application) being transferred to the user and rendered for their use." It is for creating large scale of network system architecture style. Motivation of REST was to create an architectural model how the Web service should work, such that it could serve as the guiding framework for the Web protocol's standards. REST has been applied to describe the desired Web architecture, helped to identify existing problems, to compare alternative solutions, and ensure that protocol extensions would not violate the core constraints that make the Web successful. But standards are usable even though REST is not a standard. For example:

- HTTP
- URL
- XML/JSON/HTML /etc
- Text/XML, OBJECT/JSON,TEXT/HTML etc.

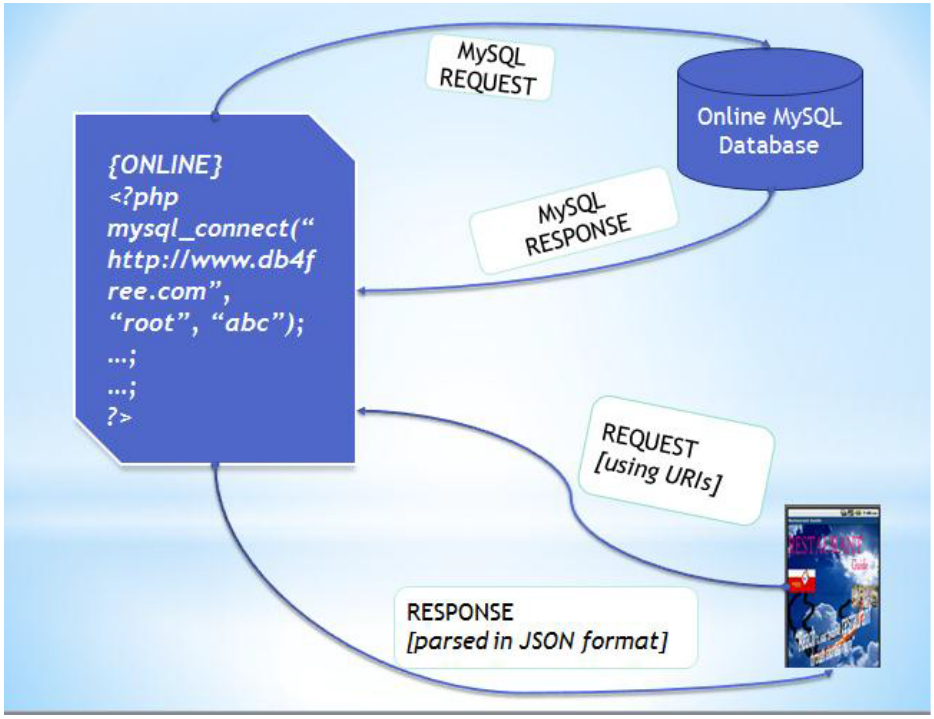


Figure 6.1: Relationships between JSON, PHP and Mysql database

6.3.6 Haversine Formula : Distance Between Two Geocodes

The haversine formula is an equation important in navigation, giving great-circle distances between two points on a sphere from their longitudes and latitudes. It is a special case of a more general formula in spherical trigonometry, the law of haversines, relating the sides and angles of spherical "triangles. On computer systems with low floating-point precision, the spherical law of cosines formula can have large rounding errors if the distance is small (if the two points are a kilometer apart on the surface of the Earth, the cosine of the central angle comes out 0.99999999). For modern 64-bit floating-point numbers, the spherical law of cosines formula, given above, does not have serious rounding errors for distances larger than a few meters on the surface of the Earth. The haversine formula is numerically better-conditioned for small distances.

$$\Delta\sigma = 2\arcsin\sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) + \cos\phi_1 \cdot \cos\phi_2 \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right)}.$$

Historically, the use of this formula was simplified by the availability of tables for the haversine function:

Although this formula is accurate for most distances on a sphere, it too suffers from rounding errors for the special (and somewhat unusual) case of antipodal points (on opposite ends of the sphere). A more complicated formula that is accurate for all distances is the following special case of the Vincenty formula for an ellipsoid with

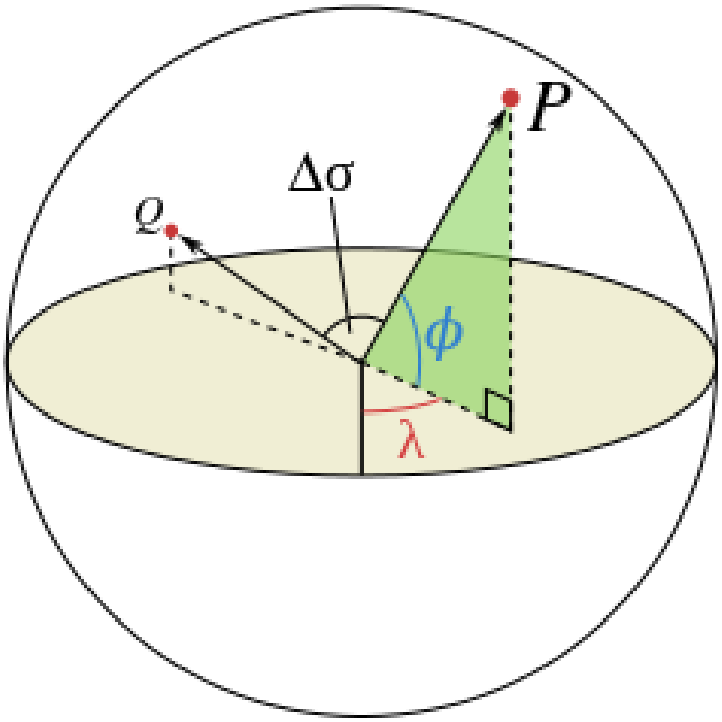


Figure 6.2: A sphere showing three geopoints

equal major and minor axes.

$$\Delta\sigma = \arctan \frac{\sqrt{(\cos\phi_2 \cdot \sin(\Delta\lambda))^2 + (\cos\phi_1 \cdot \sin\phi_2 - \sin\phi_1 \cdot \cos\phi_2 \cdot \cos(\Delta\lambda))^2}}{\sin\phi_1 \cdot \sin\phi_2 + \cos\phi_1 \cdot \cos\phi_2 \cdot \cos(\Delta\lambda)}$$

. When programming a computer, one should use the atan2() function rather than the ordinary arctangent function (atan()), so that

$$\Delta\sigma$$

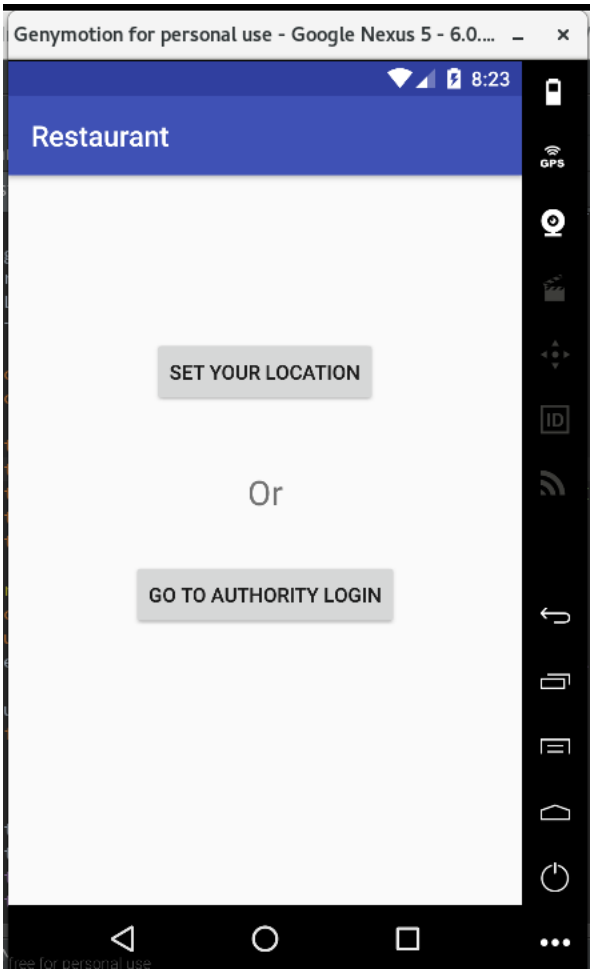
is placed in the correct quadrant.

The determination of the great-circle distance is just part of the more general problem of great-circle navigation, which also computes the azimuths at the end points and intermediate way-points.

6.3.7 RESULT AND ANALYSIS

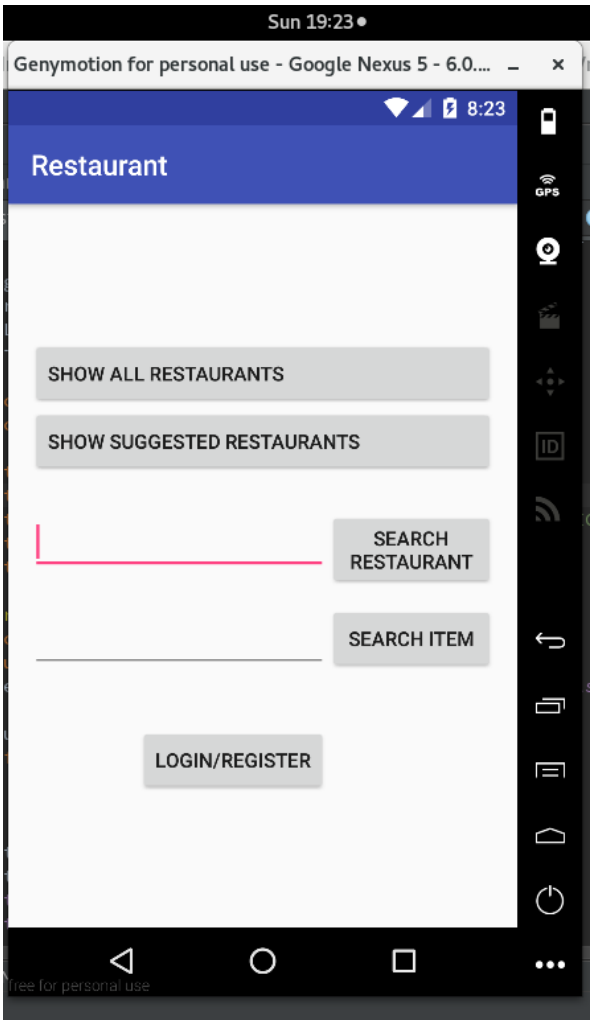
The application thus developed has the following results:

Location Detection



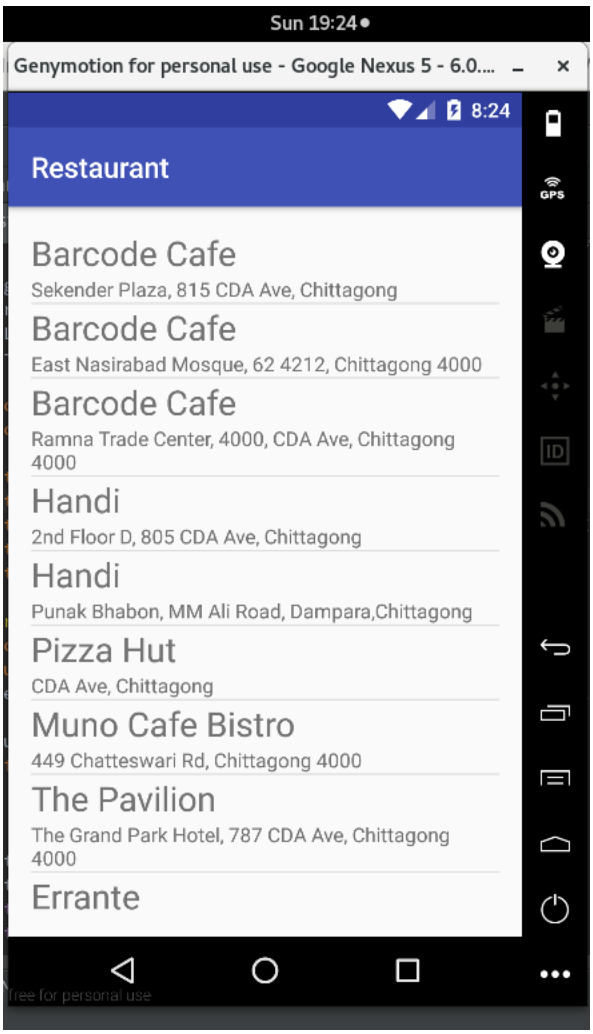
This activity tells user to choose his type - if they are users they will set their location and if they are restaurant authority they will choose their login option.

Menu



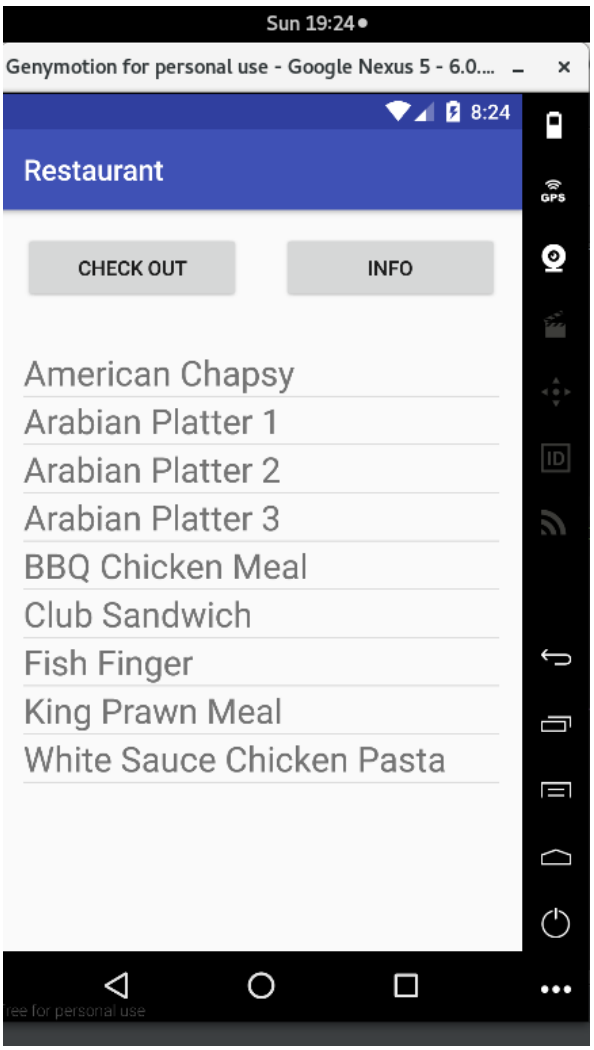
Menu allows users to choose options from - showing the entire restaurant database, showing suggested restaurant based on distance and rating, searching restaurant by name, searching item by name or login/register option.

Restaurant List



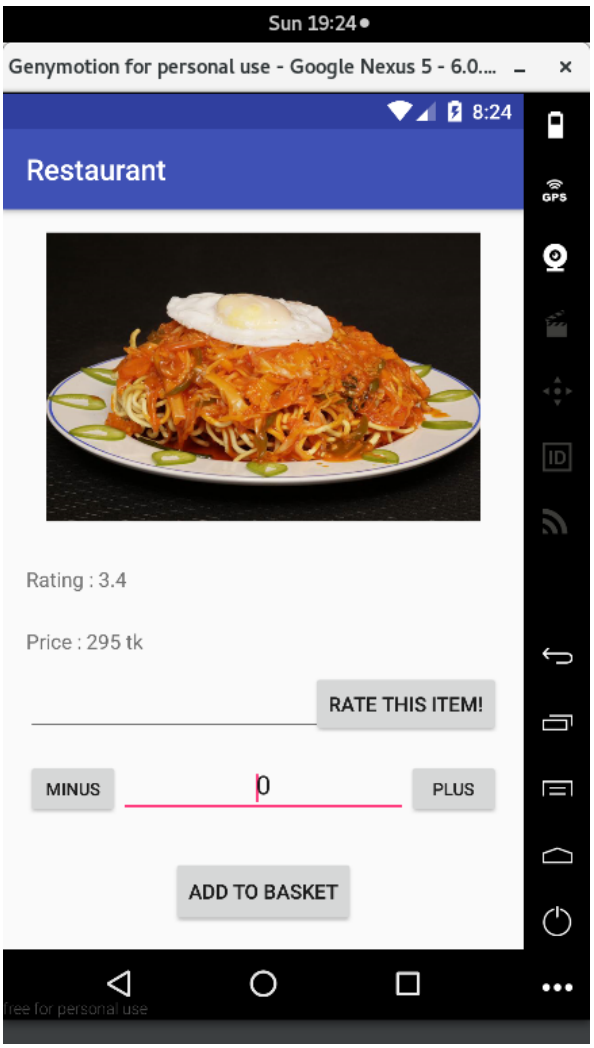
This page will show the restaurant list according to user's choice.

Restaurant Menu



If user click on any item, the app will take the user to item info.

Item Info



This activity will show image, price, rating of the item and how much user will order.

Order

User will confirm order and check out.

6.3.8 MYSQL Queries

Create Table

```
-----------------------------------------------------------------------------------------------------------+
| Restaurants | CREATE TABLE `Restaurants` (
  `TRADE_LICENSE_NO` int(11) NOT NULL AUTO_INCREMENT,
  `NAME` text,
  `RATING` float(2,1) DEFAULT NULL,
  `PAYMENT_METHOD` text,
  `START_AT` time DEFAULT NULL,
  `END_AT` time DEFAULT NULL,
  `DELIVERY_FEE` int(11) DEFAULT NULL,
  `NO_OF_USERS_RATED` int(11) DEFAULT NULL,
  PRIMARY KEY (`TRADE_LICENSE_NO`)
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=latin1 |
+-------------+--------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql> 
```

Select

```
mysql> SELECT * FROM Restaurants;
+------------------+-------------------------------------------+--------+-------------------+----------+----------+--------------+------------------
| TRADE_LICENSE_NO | NAME                                      | RATING | PAYMENT_METHOD    | START_AT | END_AT   | DELIVERY_FEE | NO_OF_USERS_RATED
+------------------+-------------------------------------------+--------+-------------------+----------+----------+--------------+------------------
|                1 | Cafe Milano                               |    3.3 | Cash On Delivery  | 02:30:00 | 02:30:00 |           50 |                30
|                2 | Errante                                   |    4.1 | Cash On Delivery  | 12:00:00 | 12:00:00 |           50 |                36
|                3 | Burgwich Town                             |    3.8 | Cash On Delivery  | 10:00:00 | 10:00:00 |           50 |                23
|                4 | Melange Coffee & Convention               |    3.7 | Cash On Delivery  | 10:00:00 | 10:00:00 |           50 |                34
|                5 | Barcode Cafe                              |    3.9 | Cash On Delivery  | 11:00:00 | 11:00:00 |           50 |               121
|                6 | Basmati Restaurant                        |    4.1 | Cash On Delivery  | 10:00:00 | 10:00:00 |           50 |                14
|                7 | The Pavilion                              |    4.1 | Cash On Delivery  | 10:00:00 | 10:00:00 |           50 |                38
|                8 | Rio Coffee                                |    4.1 | Cash On Delivery  | 10:30:00 | 10:30:00 |           50 |                 7
|                9 | Pizza Hut                                 |    4.3 | Cash On Delivery  | 11:00:00 | 11:00:00 |           50 |                64
|               10 | Avalon                                    |    3.7 | Cash On Delivery  | 12:00:00 | 12:00:00 |           50 |                25
|               11 | Greedy Guts                               |    3.7 | Cash On Delivery  | 10:30:00 | 10:30:00 |           50 |                26
|               12 | Rasoi                                     |    3.7 | Cash On Delivery  | 12:00:00 | 12:00:00 |           50 |                22
|               13 | Impala Chinese & South Indian Restaurant  |    4.1 | Cash On Delivery  | 11:30:00 | 11:30:00 |           50 |                21
|               14 | Muno Cafe Bistro                          |    3.6 | Cash On Delivery  | 02:00:00 | 02:00:00 |           50 |                44
|               15 | Handi                                     |    4.0 | Cash On Delivery  | 10:00:00 | 10:00:00 |           50 |               115
+------------------+-------------------------------------------+--------+-------------------+----------+----------+--------------+------------------
15 rows in set (0.00 sec)

mysql>
mysql> 
```

Insert

```
mysql> INSERT INTO Restaurants(NAME, RATING, START_AT, END_AT, DELIVERY_FEE, NO_OF_USERS_RATED) VALUES('Crush Cafe', '4.5', '00:09:00', '12:00:00'
, 45);
Query OK, 1 row affected (0.04 sec)

mysql> SELECT * FROM Restaurants;
+-----------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
| TRADE_LICENSE_NO | NAME                                    | RATING | START_AT | END_AT   | DELIVERY_FEE | NO_OF_USERS_RATED |
+-----------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
|               1 | Cafe Milano                             |    3.3 | 02:30:00 | 02:30:00 |           50 |                30 |
|               2 | Errante                                 |    4.1 | 12:00:00 | 12:00:00 |           50 |                36 |
|               3 | Burgwich Town                           |    3.8 | 10:00:00 | 10:00:00 |           50 |                23 |
|               4 | Melange Coffee & Convention             |    3.7 | 10:00:00 | 10:00:00 |           50 |                34 |
|               5 | Barcode Cafe                            |    3.9 | 11:00:00 | 11:00:00 |           50 |               121 |
|               6 | Basmati Restaurant                      |    4.1 | 10:00:00 | 10:00:00 |           50 |                14 |
|               7 | The Pavilion                            |    4.1 | 10:00:00 | 10:00:00 |           50 |                38 |
|               8 | Rio Coffee                              |    4.1 | 10:30:00 | 10:30:00 |           50 |                 7 |
|               9 | Pizza Hut                               |    4.3 | 11:00:00 | 11:00:00 |           50 |                64 |
|              10 | Avalon                                  |    3.7 | 12:00:00 | 12:00:00 |           50 |                25 |
|              11 | Greedy Guts                             |    3.7 | 10:30:00 | 10:30:00 |           50 |                26 |
|              12 | Rasoi                                   |    3.7 | 12:00:00 | 12:00:00 |           50 |                22 |
|              13 | Impala Chinese & South Indian Restaurant |    4.1 | 11:30:00 | 11:30:00 |           50 |                21 |
|              14 | Muno Cafe Bistro                        |    3.6 | 02:00:00 | 02:00:00 |           50 |                44 |
|              15 | Handi                                   |    4.0 | 10:00:00 | 10:00:00 |           50 |               115 |
|              16 | Crush Cafe                              |    4.5 | 00:09:00 | 12:00:00 |           50 |                45 |
+-----------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
16 rows in set (0.00 sec)

mysql>
```

Delete

```
mysql> DELETE FROM Restaurants WHERE TRADE_LICENSE_NO = 16;
Query OK, 1 row affected (0.07 sec)

mysql> SELECT * FROM Restaurants;
+-----------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
| TRADE_LICENSE_NO | NAME                                    | RATING | START_AT | END_AT   | DELIVERY_FEE | NO_OF_USERS_RATED |
+-----------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
|               1 | Cafe Milano                             |    3.3 | 02:30:00 | 02:30:00 |           50 |                30 |
|               2 | Errante                                 |    4.1 | 12:00:00 | 12:00:00 |           50 |                36 |
|               3 | Burgwich Town                           |    3.8 | 10:00:00 | 10:00:00 |           50 |                23 |
|               4 | Melange Coffee & Convention             |    3.7 | 10:00:00 | 10:00:00 |           50 |                34 |
|               5 | Barcode Cafe                            |    3.9 | 11:00:00 | 11:00:00 |           50 |               121 |
|               6 | Basmati Restaurant                      |    4.1 | 10:00:00 | 10:00:00 |           50 |                14 |
|               7 | The Pavilion                            |    4.1 | 10:00:00 | 10:00:00 |           50 |                38 |
|               8 | Rio Coffee                              |    4.1 | 10:30:00 | 10:30:00 |           50 |                 7 |
|               9 | Pizza Hut                               |    4.3 | 11:00:00 | 11:00:00 |           50 |                64 |
|              10 | Avalon                                  |    3.7 | 12:00:00 | 12:00:00 |           50 |                25 |
|              11 | Greedy Guts                             |    3.7 | 10:30:00 | 10:30:00 |           50 |                26 |
|              12 | Rasoi                                   |    3.7 | 12:00:00 | 12:00:00 |           50 |                22 |
|              13 | Impala Chinese & South Indian Restaurant |    4.1 | 11:30:00 | 11:30:00 |           50 |                21 |
|              14 | Muno Cafe Bistro                        |    3.6 | 02:00:00 | 02:00:00 |           50 |                44 |
|              15 | Handi                                   |    4.0 | 10:00:00 | 10:00:00 |           50 |               115 |
+-----------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
15 rows in set (0.00 sec)

mysql>
```

Update

```
mysql> UPDATE Restaurants SET RATING = '4.5' WHERE TRADE_LICENSE_NO = 1;
Query OK, 1 row affected (0.06 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Restaurants;
+------------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
| TRADE_LICENSE_NO | NAME                                    | RATING | START_AT | END_AT   | DELIVERY_FEE | NO_OF_USERS_RATED |
+------------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
|                1 | Cafe Milano                             |    4.5 | 02:30:00 | 02:30:00 |           50 |                30 |
|                2 | Errante                                 |    4.1 | 12:00:00 | 12:00:00 |           50 |                36 |
|                3 | Burgwich Town                           |    3.8 | 10:00:00 | 10:00:00 |           50 |                23 |
|                4 | Melange Coffee & Convention             |    3.7 | 10:00:00 | 10:00:00 |           50 |                34 |
|                5 | Barcode Cafe                            |    3.9 | 11:00:00 | 11:00:00 |           50 |               121 |
|                6 | Basmati Restaurant                      |    4.1 | 10:00:00 | 10:00:00 |           50 |                14 |
|                7 | The Pavilion                            |    4.1 | 10:00:00 | 10:00:00 |           50 |                38 |
|                8 | Rio Coffee                              |    4.1 | 10:30:00 | 10:30:00 |           50 |                 7 |
|                9 | Pizza Hut                               |    4.3 | 11:00:00 | 11:00:00 |           50 |                64 |
|               10 | Avalon                                  |    3.7 | 12:00:00 | 12:00:00 |           50 |                25 |
|               11 | Greedy Guts                             |    3.7 | 10:30:00 | 10:30:00 |           50 |                26 |
|               12 | Rasoi                                   |    3.7 | 12:00:00 | 12:00:00 |           50 |                22 |
|               13 | Impala Chinese & South Indian Restaurant |    4.1 | 11:30:00 | 11:30:00 |           50 |                21 |
|               14 | Muno Cafe Bistro                        |    3.6 | 02:00:00 | 02:00:00 |           50 |                44 |
|               15 | Handi                                   |    4.0 | 10:00:00 | 10:00:00 |           50 |               115 |
+------------------+-----------------------------------------+--------+----------+----------+--------------+-------------------+
15 rows in set (0.00 sec)

mysql>
```

Alter

```
mysql> ALTER TABLE Restaurants DROP COLUMN PAYMENT_METHOD;
Query OK, 0 rows affected (0.63 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> select * from Restaurants;
+------------------+-------------+--------+----------+----------+--------------+-------------------+
| TRADE_LICENSE_NO | NAME        | RATING | START_AT | END_AT   | DELIVERY_FEE | NO_OF_USERS_RATED |
+------------------+-------------+--------+----------+----------+--------------+-------------------+
|                1 | Barcode Cafe |    4.5 | NULL     | NULL     |           50 |                 1 |
|                2 | Pizza Hut    |    4.8 | NULL     | NULL     |           50 |                 0 |
|                3 | Pavilion     |    4.8 | NULL     | NULL     |           50 |                 0 |
|                4 | Impala       |    4.5 | NULL     | NULL     |           50 |                 0 |
|                5 | Cafe Milange |    4.7 | NULL     | NULL     |           50 |                 0 |
|                6 | Cafe Milano  |    4.7 | NULL     | NULL     |           50 |                 3 |
|                7 | Muno Cafe    |    4.3 | NULL     | NULL     |           50 |                 0 |
|                8 | Avalon       |    4.4 | NULL     | NULL     |           50 |                 0 |
|                9 | Meridian     |    4.2 | NULL     | NULL     |           50 |                 0 |
|               10 | Baburchi     |    4.1 | NULL     | NULL     |           50 |                 0 |
+------------------+-------------+--------+----------+----------+--------------+-------------------+
10 rows in set (0.00 sec)

mysql>
```

Chapter 7

Future Recommendation

The heart of the entire ordering system is the Database. Currently the system is only available for small scale restaurants. For Large restaurants, performance considerations should be taken into account in terms of Hardware/Software capacity/Page load time etc. Also, security vulnerabilities should be evaluated for large scale systems. In future this can also be available as a Mobile application and can be integrated with in store Touch Screen Order devices. I am also certain that if this system goes into actual use, many requests will arise for additional features which I had not previously considered, but would be useful to have. For this reason, I feel as though the application can be constantly evolving, which I consider a very good thing. As we have some of the limitation in our application, we can enhance our application in it. We can enhance the application to add more and more hotels and restaurant available within the valley. Furthermore we need to make our application check the available balance of the user. We should deny the request of the delivery to the user if the balance is below the certain balance. In the upcoming days we also can enhance this application in the area of Google Map API . The Google map shows only one path between the user and destination. We can show more than one path with their distance so that user can select any route which can be easy for them. The following section describes the work that will be implemented with future releases of the software.

- Customize orders: Allow customers to customize food orders
- Enhance User Interface by adding more user interactive features. Provide Deals and promotional Offer details to home page. Provide Recipes of the Week/Day to Home Page
- Payment Options: Add different payment options such as PayPal, Cash, Gift Cards etc. Allow to save payment details for future use.
- Allow to process an order as a Guest
- Delivery Options: Add delivery option
- Order Process Estimate: Provide customer a visual graphical order status bar
- Order Status: Show only Active orders to Restaurant Employees.

- Order Ready notification: Send an Order Ready notification to the customer
- Integrate with In store touch screen devices like iPad

7.1 References

1. Reto Meier, Professional Android Application Development, Wiley Publishing Inc., Indianapolis, Indiana, 2009.
2. Crockford, Douglas, "Introducing JSON, May 28, 2009.
3. JSON in Java , http://www.json.org.
4. Mark Murphy , The Busy Coder's Guide to Android Development Commons Ware,2009.
5. www.w3school.com
6. Modern Database Management-By Jeffrey A.Hoffer
7. Database System Concepts- By Henry F.Korth
8. Wikipidia